

Inleiding tot Matlab

Brecht Baeten

2 oktober 2015

1 Matlab voor ingenieurs

De broncode van deze tekst alsook alle voorbeelden kunnen worden gedownload via <https://github.com/BrechtBa/inleiding-tot-matlab/>.

2 Command line

Wanneer Matlab gestart is kan je rechtstreeks commando's uitvoeren in de command line interface. Typ hier bijvoorbeeld:

```
disp('Hello World!')
```

3 Scripts

Omdat het telkens opnieuw invoeren van commando's achter elkaar niet zo praktisch is, is het interessant om een reeks commando's op te slaan in een script en dit uit te voeren. Matlab code kan opgeslagen worden in tekst bestanden met de extensie ".m". Binnen de Matlab IDE zit een handige text editor specifiek ontworpen voor het schrijven van m-files. Open een de m-file editor en typ "**disp**('Hello World!')" en sla het bestand op als `hello_world.m`. Om het bestand uit te voeren kan je de naam van het bestand typen in de command line interface of op de run knop in de werkbalk van de editor klikken. Het resultaat is hetzelfde als daarnet.

Python heeft een aantal ingebouwde variabele types. Open een nieuw bestand, typ onderstaande commando's en sla het op als `variables.m` en voer het uit.

```
# float
A = 1;
A

B = 3.572;
B

# string
C = 'string';
C

# vector
```

```

D = [1,2,3];
D
D(1)
D(end)

# cell
E = {1, 'test', 4, D};
E

# structure
F = struct('value',1, 'name','test', 'spam','eggs');
F
F.spam

```

Alle reeds gedefinieerde variabelen zijn nu ook beschikbaar in de console. Typ bijvoorbeeld:

```
E[3]
```

of:

```
A+B
```

De grote kracht van Matlab zit in het werken met vectoren en matrices (Matlab komt van MA-Trix LABoratory). Vectoren kunnen enkel waarden van hetzelfde type bevatten. Een *Cell array* daarentegen kan waarden van een verschillend type bevatten. De Matlab *Struct* variabele is zeer interessant omdat deze kan gebruikt worden om verschillende soorten data gestructureerd te groeperen. In Matlab wordt elke variabele door zijn waarde gedefinieerd.

Scripts kunnen in Matlab alleen uitgevoerd worden indien ze in de Matlab current folder staan of als ze in een folder die tot het matlab zoekpad behoort staan. Met behulp van het *Current Folder* venster kan je binnen Matlab browsen naar een andere folder, dit kan ook met het "**cd()**" commando. Met behulp van het "**addpath()**" commando kan je een bepaalde folder, waarin bijvoorbeeld een aantal veel voorkomende functies staan, aan het matlab path toevoegen. Zo worden deze functies beschikbaar.

Echt programmeren begint pas bij maken van lussen en voorwaarden: flow control. In Matlab is dit zeer eenvoudig met behulp van "**for**" lussen of "**if else**" structuren:

```

for i=1:10
    if mod(i,2) == 0
        disp(sprintf('%i is even',i))
    else
        disp(sprintf('%i is oneven',i))
    end
end

```

In Matlab moet elke "**for**" lussen of "**if else**" constructie eindigen op een "**end**" commando. Het "**1:10**" commando in de "**for**" lus maakt een *Vector* met waarden van 0 tot en met 9. De "**sprintf('%f is even',i)**" functie formatteert zijn argumenten op de plaats van de "**%f**" in de format string al floating point numbers. Er zijn verschillende mogelijkheden om getallen te formateren, even zoeken in de help functie leert je heel veel over zulke ingebouwde functies.

4 Functies

Een set commando's die vaak op dezelfde manier gebruikt worden kan je groeperen in een functie. Een functie definieer je met het "**function**" keyword, gevolgd door de output argumenten, een gelijkheidsteken, de functie naam en de argument namen tussen haakjes op het einde van een functie hoort zoals gewoonlijk een "**end**". Functies moeten binnen Matlab in een apparte m-file met dezelfde naam als de functie worden opgeslagen.

```
function val = spam(A,B,C,D)
    val = 100*A+10*B+C+0.1*D
end
```

Dit werk is gelicenseerd onder de licentie Creative Commons Naamsvermelding-GelijkDelen 4.0 Internationaal. Ga naar <http://creativecommons.org/licenses/by-sa/4.0/> om een kopie van de licentie te kunnen lezen.